Reinforcement Learning with Classifier Selection for Focused Crawling

Ioannis Partalas,[1] George Paliouras,[2] Ioannis Vlahavas[1]

Abstract. Focused crawlers are programs that wander in the Web, using its graph structure, and gather pages that belong to a specific topic. The most critical task in Focused Crawling is the scoring of the URLs as it designates the path that the crawler will follow, and thus its effectiveness. In this paper we propose a novel scheme for assigning scores to the URLs, based on the Reinforcement Learning (RL) framework. The proposed approach learns to select the best classifier for ordering the URLs. This formulation reduces the size of the search space for the RL method and makes the problem tractable. We evaluate the proposed approach on-line on a number of topics, which offers a realistic view of its performance, comparing it also with a RL method and a simple but effective classifier-based crawler. The results demonstrate the strength of the proposed approach.

1 Introduction

In this paper we propose a novel adaptive focused crawler that is based on the RL framework [5]. More specifically, RL is employed for selecting an appropriate classifier that will in turn evaluate the links that the crawler must follow. The introduction of link classifiers reduces the size of the search space for the RL method and makes the problem tractable. We evaluate the proposed approach on a number of topics, comparing it with an RL approach from the bibliography and a classifier-based crawler. The results demonstrate the robustness and the efficiency of the proposed approach.

2 Reinforcement Learning with Classifier Selection

In this work we propose an adaptive approach, dubbed Reinforcement Learning with Classifier Selection (RLwCS), to evaluate URLs, based on the RL framework. RLwCS maintains a pool of classifiers, $H = \{h_1, \dots, h_k\}$, that can be used for URL evaluation, and seeks a policy for selecting the best classifier, h_t, for a page to perform the evaluation task. In other words, the crawler must select dynamically a classifier for each page, according to the characteristics of the page. We solve this problem using an RL approach. In our case, there are just two classes, as a URL or page can be relevant or not to a specific topic.

We represent the problem of selecting a classifier for evaluating the URLs, as an RL process. The state is defined as the page that is currently retrieved by the agent, on the basis that the perception of the environment arises mainly by the pages retrieved at any given time. Actions are the different classifiers, $h_t \in H$. We add an extra action which is denoted as S and combines the classifiers in a majority scheme. The set of actions is thus $H \cup \{S\}$. The state transitions are deterministic as the probability of moving to a page when selecting a classifier for evaluation is equal to 1. The selected classifier is the one that scores the URLs of a visited page. More specifically, the classifier's score that a URL belongs to the relevant class.

The reward for selecting a classifier depends on the relevance of the page that the crawler visit. If the page is relevant, the reward is 1, while otherwise the reward is 0. Thus, we seek to find an optimal policy for mapping pages to classifiers in order to maximize the accumulated reward received over time.

The mechanism that is used for training the RL module is the *Q-learning* algorithm [6]. Q-learning finds an optimal policy based on the *action-value function*, $Q(s, a)$. The Q function expresses the benefit of following the action a when in state s. In our case the value of selecting a classifier in a specific page is associated with the expected relevance of the next page (state) that the crawler will fetch.

Next, we need to define the features that will be used to represent both the states and the actions. Based on the literature of focused crawling we chose the following features to represent a state-action pair:

- Relevance score of a page with respect to the specific domain.
- Relevance score of the page, computed by the selected classifier (action).
- Average relevance score of the parents of the page that is crawled.
- Hub score.

We employ function approximation to tackle the problem of the large state-action space. A well-known method is the combination of Q-learning with eligibility traces, $Q(\lambda)$, and gradient descent function approximation [5]. Additionally, linear methods are used to approximate and represent the value function. Further details about the function approximation algorithm that we used can be found in [5].

3 Experimental Setup

We constructed a number of topic-specific datasets following the procedure that is described in [2]. Table 1 shows the topics that we selected for experimentation[3]. For each topic's URL we downloaded the corresponding page and constructed the instances based on the textual information. More specifically, for each document downloaded we produced the TF-IDF vectors using the weighted scheme proposed by Salton and Buckley [3]. Each instance of the on-topic and off-topic documents is named *relevant* or *irrelevant* respectively.[4]

[1] Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece, email: {partalas,vlahavas}@csd.auth.gr
[2] Institute of Informatics and Telecommunications, National Centre for Scientific Research "Demokritos", email: paliourg@itt.demokritos.gr

[3] http://dmoz.org
[4] The datasets created are available at http://mlkd.csd.auth.gr/fcrawling.html

Table 1. ODP topics.

Topic	Number of URLs
Shopping/Auctions/Antiques_and_Collectibles/	62
Health/Medicine/Osteopathy/	166
Games/Video_Games/Puzzle/Tetris-like/	72
News/Weather/Air_Quality/	114
Science/Astronomy/Amateur/Astrophotography _and_CCD_Imaging/	196
Health/Medicine/Informatics/Telemedicine/	64
Sports/Winter_Sports/Snowboarding/	179
Sports/Hockey/Ice_Hockey/	239
Arts/Literature/Periods_and_Movements/	275
Health/Alternative/Aromatherapy/	103

After creating the set of relevant and irrelevant instances we train the classifiers for each topic that will form the action set for RLwCS, with the addition of the extra action that combines the opinions of the classifiers using the majority scheme. For an instance x the output of the majority scheme is $S(x) = \max_{c_j} \sum_{m=1}^{k} h_m(x, c_j)$, where h_m outputs a probability distribution for each class $c_j, j = 1 \dots n$..We trained four classifiers using the WEKA machine learning library [8]:

- Neural network (NN): 16 hidden nodes and learning rate 0.4.
- Support vector machine (SVM): polynomial kernel with degree 1.
- Naive Bayes (NB): with kernel estimation.
- Decision tree (DT): with Laplace smoothing and reduced error pruning.

The proposed approach, RLwCS, is compared with a base crawler that uses a SVM to assign scores to the URLs and with Temporal Difference Focused Crawling (TD-FC) method [1].

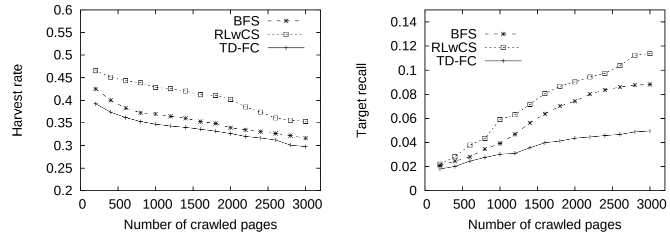
The experiments for the crawlers are performed on-line in order to obtain a realistic estimate of their performance. We must note here that the majority of the approaches reported in the literature, conducted their experiments offline in a managed environment. The online evaluation in a variety of topics allow us to make more accurate statistical tests in order to detect significant differences in the performances of the crawlers.

For the purposes of evaluation, we used two analogous metrics to the well-known precision and recall, that is *harvest rate* and *target recall* [4].

4 Results and Discussion

Figure 1(a) presents the average harvest rate of each algorithm for all topics, against the number of crawled pages. We first notice that RLwCS clearly outperforms both BFS anf TD-FC, as it manages to collect more relevant pages. In order to investigate whether the performance differences between RLwCS and the other two algorithms are significant, we use the Wilcoxon signed rank test [7]. We performed 2 tests, one for each paired comparison of RLwCS with each of the other algorithms on each topic, at a confidence level of 95%. The test was performed on various points during the crawling process, and more specifically per 200 crawled pages. The test found that RLwCS is significantly better than all the other algorithms during the whole crawling process (200 to 3000 pages) on all topics.

Another interesting observation is the fact that the proposed approach achieves a high harvest rate in the first 200 pages which is a strong advantage in on-line crawling tasks where the crawler must gather relevant pages in a small time frame and a small number of visited pages. Figure 1(b) shows the target recall curves for the competing algorithms, averaged across all topics. We notice again that



(a) Average harvest rate (b) Average target reacll

Figure 1. Average harvest rate and target recall.

the proposed approach obtains the highest values during the crawling process and outperforms the other two methods. Wilcoxon tests at a confidence level of 95% report significant differences only after the first 600 pages have been crawled. This is again a very encouraging result for the proposed approach.

5 Conclusions

In this paper we presented a novel Focused Crawling approach, named RLwCS, which is based on the Reinforcement Learning framework. The crawler learns to select an appropriate classifier for ordering the URLs of each Web page that it visits. We compared the proposed approach with the well-known Best-First Search crawler and a pure RL approach, on a number of topic-specific datasets. The crawlers were tested on-line, in order to obtain realistic measurements of their performance. The analysis of the results led to several interesting conclusions. The proposed approach manages to achieve good performance outperforming the BFS which is considered in the literature as a very effective crawler.

Acknowledgments

We would like to thank Ioannis Katakis for providing us with the source code for text processing and Michalis Lagoudakis for interesting discussions that led to this work.

REFERENCES

[1] A. Grigoriadis and G. Paliouras, ‘Focused crawling using temporal difference-learning’, in *Proc. 3th Hellenic Conference on Artificial Intelligence*, pp. 142–153, (2004).

[2] Gautam Pant and Padmini Srinivasan, ‘Learning to crawl: Comparing classification schemes’, *ACM Transanction on Information Systems*, **23**(4), 430–462, (2005).

[3] Gerard Salton and Christopher Buckley, ‘Term-weighting approaches in automatic text retrieval’, *Information Processing and Management*, **24**(5), 513–523, (1988).

[4] P. Srinivasan, F. Menczer, and G. Pant, ‘A general evaluation framework for topical crawlers’, *Information Retrieval*, **8**(3), 417–447, (2005).

[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning, An Introduction*, MIT Press, 1999.

[6] C.J. Watkins and P. Dayan, ‘Q-learning’, *Machine Learning*, **8**, 279–292, (1992).

[7] F. Wilcoxon, ‘Individual comparisons by ranking methods’, *Biometrics*, **1**, 80–83, (1945).

[8] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques, 2nd Edition*, Morgan Kaufmann, 2005.